**TM Forum Specification**

**Service Activation and Configuration API REST Specification**

**TMF640**

**Release 18.5.0**

**January 2019**

|  |  |
| --- | --- |
| **Latest Update: TM Forum Release 18.5.0** | **Member Evaluation** |
| **Version 4.0.0** | **IPR Mode: RAND** |

# NOTICE

Copyright © TM Forum 2019. All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to TM FORUM, except as needed for the purpose of developing any document or deliverable produced by a TM FORUM Collaboration Project Team (in which case the rules applicable to copyrights, as set forth in the [TM FORUM IPR Policy](http://www.tmforum.org/IPRPolicy/11525/home.html), must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by TM FORUM or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and TM FORUM DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Direct inquiries to the TM Forum office:

4 Century Drive, Suite 100

Parsippany, NJ 07054, USA

Tel No. +1 973 944 5100

Fax No. +1 973 998 7916

TM Forum Web Page: [www.tmforum.org](http://www.tmforum.org/)

# Table of Contents

[NOTICE 2](#_Toc514836422)

[Table of Contents 3](#_Toc514836423)

[List of Tables 4](#_Toc514836424)

[Introduction 5](#_Toc514836425)

[SAMPLE USE CASES 6](#_Toc514836426)

[Support of polymorphism and extension patterns 7](#_Toc514836427)

[RESOURCE MODEL 8](#_Toc514836428)

[Managed Entity and Task Resource Models 8](#_Toc514836429)

[FIRST resource 8](#_Toc514836430)

[Notification Resource Models 8](#_Toc514836431)

[First Notification 8](#_Toc514836432)

[API OPERATIONS 9](#_Toc514836433)

[VERB url 9](#_Toc514836434)

[API NOTIFICATIONS 10](#_Toc514836435)

[Register listener 10](#_Toc514836436)

[Unregister listener 11](#_Toc514836437)

[Publish Event to listener 11](#_Toc514836438)

[Acknowledgements 13](#_Toc514836439)

[Release History 13](#_Toc514836440)

[Contributors to Document 13](#_Toc514836441)

# LIST OF TABLES

N/A

# INTRODUCTION

Services may be activated or configured by the Activation and Configuration API.

The following examples use the characteristic specification and the strongly typed patterns.

1) Using the characteristic specification pattern. When the characteristic specification pattern is used it is assumed that a service specification with the corresponding characteristic specifications exist.

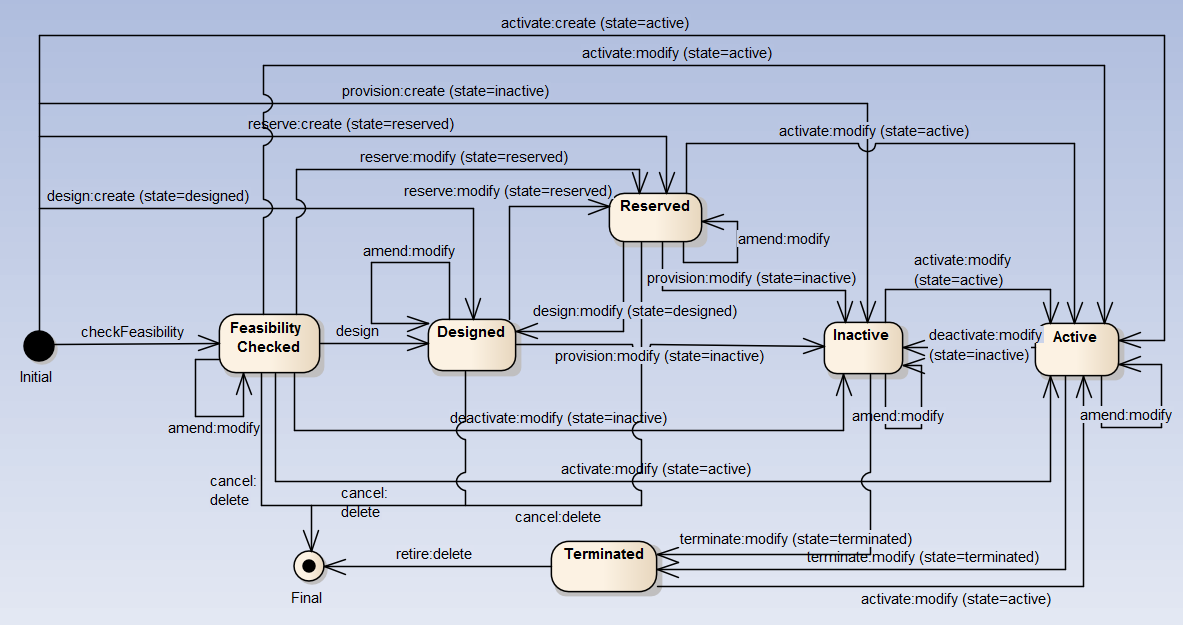
|  |
| --- |
| "id" : "id1234567890",  "href" : "http//..",  "state" : "active",  "serviceSpecification":{  "id":"conferenceBridgeEquipment",  "href":"http: //serverlocation:port/catalogManagement/serviceSpecification /conferenceBridgeEquipment"  },  "characteristic":[  {  "name":"numberOfVc500Units",  "valueType": "integer",  "value":"1"  },  {  "name":"numberOfVc100Units",  "valueType": "integer",  "value":"2"  },  {  "name":"routerType",  "valueType": "string",  "value":"CiscoASR1000"  },  {  "name":"powerSupply",  "valueType": "string",  "value":"UK"  }  ]}  } |

2) Using the Strongly typed pattern. When the strongly typed pattern is used it is assumed that a corresponding YAML or JSON schema is defined for the Service.

|  |
| --- |
| {  "id" : "id1234567890",  "href" : "http//..",  "state" : "active",  "@type" : "conferenceBridgeEquipment",  "@schemaLocation" : "http://../conferenceBridgeEquipment.json",  "serviceSpecification":{  "id":"conferenceBridgeEquipment",  "href":"http: //serverlocation:port/catalogManagement/serviceSpecification /conferenceBridgeEquipment"  },  "numberOfVc500Units": "1",  "numberOfVc100Units": "2",  "routerType": "CiscoASR1000",  "powerSupply": "UK"  }  } |

## Lifecycle of Service

Here is the state machine diagram for a Service.



# Support of polymorphism and extension patterns

Support of polymorphic collections and types and schema based extension is provided by means of a list of generic meta-attributes that we describe below. Polymorphism in collections occurs when entities inherit from base entities, for instance a TypeAService or TypeBService inheriting properties from the base Service entity.

Generic support of polymorphism and pattern extensions is described in the TMF API Guidelines v3.0 Part 2 document.

The @type attribute provides a way to represent the actual class type of an entity. For example, within a list of Service instances some may be instances of TypeAService where other could be instances of TypeBService. The @type gives this information. All resources and sub-resources of this API have a @type attributes that can be provided when this is useful.

The @referredType can be used within reference entities (like for instance a RelatedParty object) to explicitly denote the actual entity type of the referred class. Notice that in reference entities the @type, when used, denotes the class type of the reference itself, such as RelatedParty, and not the class type of the referred object. However, since reference classes are rarely sub-classed, @type is generally not useful in reference objects.

The @schemaLocation property can be used in resources to allow specifying user-defined properties of an Entity or to specify the expected characteristics of an entity.

The @baseType attribute gives a way to provide explicitly the base of class of a given resource that has been extended.

# RESOURCE MODEL

## Managed Entity and Task Resource Models

### FIRST resource

## Notification Resource Models

### First Notification

# API OPERATIONS

Remember the following Uniform Contract:

|  |  |  |
| --- | --- | --- |
| Operation on Entities | Uniform API Operation | Description |
| Query Entities | GET Resource | GET must be used to retrieve a representation of a resource. |
| Create Entity | POST Resource | POST must be used to create a new resource |
| Partial Update of an Entity | PATCH Resource | PATCH must be used to partially update a resource |
| Complete Update of an Entity | PUT Resource | PUT must be used to completely update a resource identified by its resource URI |
| Remove an Entity | DELETE Resource | DELETE must be used to remove a resource |
| Execute an Action on an Entity | POST on TASK Resource | POST must be used to execute Task Resources |
| Other Request Methods | POST on TASK Resource | GET and POST must not be used to tunnel other request methods. |

Filtering and attribute selection rules are described in the TMF REST Design Guidelines Part 1 document.

Notifications are also described in a subsequent section.

## VERB url

# API NOTIFICATIONS

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines part 1. Refer to the guidelines for more details.

## Register listener

**POST /hub**

**Description**

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

**Behavior**

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

**Usage Samples**

Here's an example of a request for registering a listener.

|  |
| --- |
| **Request** |
| POST /api/hub  Accept: application/json  {"callback": "http://in.listener.com"} |
| **Response** |
| 201  Content-Type: application/json  Location: /api/hub/42  {"id":"42","callback":"http://in.listener.com","query":null} |

## Unregister listener

**DELETE /hub/{id}**

**Description**

Clears the communication endpoint address that was set by creating the Hub.

**Behavior**

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

**Usage Samples**

Here's an example of a request for un-registering a listener.

|  |
| --- |
| **Request** |
| DELETE /api/hub/42  Accept: application/json |
| **Response** |
| 204 |

## Publish Event to listener

**POST /client/listener**

**Description**

Clears the communication endpoint address that was set by creating the Hub.

Provides to a registered listener the description of the event that was raised. The /client/listener url is the callback url passed when registering the listener.

**Behavior**

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

**Usage Samples**

Here's an example of a notification received by the listener. In this example “EVENT TYPE” should be replaced by one of the notification types supported by this API (see Notification resources Models section) and EVENT BODY refers to the data structure of the given notification type.

|  |
| --- |
| **Request** |
| POST /client/listener  Accept: application/json  {  "event": {  EVENT BODY  },  "eventType": "EVENT\_TYPE"  } |
| **Response** |
| 201 |

For detailed examples on the general TM Forum notification mechanism, see the TMF REST Design Guidelines Part 1 document.

# Acknowledgements

## Version History

|  |  |  |  |
| --- | --- | --- | --- |
| **Release Number** | **Date** | **Release led by:** | **Description** |
| Version 1.0.0 | 12 Nov 2015 | Pierre Gauthier  TM Forum | Submitted for Fx15.5 |
| Version 1.0.1 | 12 Nov 2015 | Alicja Kawecki  TM Forum | Updated cover, header; minor formatting/style corrections prior to publishing |
| Version 1.0.2 | 02 May 2016 | Alicja Kawecki,  TM Forum | Updated cover, footer, Notice to reflect TM Forum Approved status |
| Version 4.0.0 | 14 Jan 2019 | Jonathan Goldberg  Amdocs  [Jonathan.Goldberg@amdocs.com](mailto:Jonathan.Goldberg@amdocs.com) | Schema alignment for NaaS APIs |

## Release History

|  |  |  |  |
| --- | --- | --- | --- |
| **Release Number** | **Date** | **Release led by:** | **Description** |
| 15.5.1 | 02 May 2016 | Alicja Kawecki,  TM Forum | Updated to reflect TM Forum Approved Status |
| 18.5.0 | 14 Jan 2019 | Jonathan Goldberg  Amdocs  [Jonathan.Goldberg@amdocs.com](mailto:Jonathan.Goldberg@amdocs.com) | Schema alignment for NaaS APIs |

## Contributors to Document

|  |  |
| --- | --- |
| Andrew Forth | Amdocs |
| August-Wilhelm Jagau | Ericsson |
| Pierre Gauthier | TM Forum |
| Jonathan Goldberg | Amdocs |